View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Purdue E-Pubs

Purdue University

Purdue e-Pubs

Department of Computer Science Technical Reports

Department of Computer Science

1991

# The SHASTRA Distributed and Collaborative Geometric Design Environment

Vinod Anupam

Chanderjit L. Bajaj

Andrew V. Royappa

Report Number:
91-075

Anupam, Vinod; Bajaj, Chanderjit L.; and Royappa, Andrew V., "The SHASTRA Distributed and Collaborative Geometric Design Environment" (1991). *Department of Computer Science Technical Reports.* Paper 914.
https://docs.lib.purdue.edu/cstech/914

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries.
Please contact epubs@purdue.edu for additional information.

# THE SHASTRA DISTRIBUTED COLLABORATIVBE GEOMETRIC DESIGN ENVIRONMENT

**Vinod Anupam**
**Chanderjit L. Bajaj**
**Andrew V. Royappa**

**CSD-TR-91-075**
**October 1991**

# The SHASTRA Distributed and Collaborative Geometric Design Environment*

*Vinod Anupam* *Chanderjit L. Bajaj* *Andrew V. Royappa*

Department of Computer Science
Purdue University
West Lafayette, IN 47907

**Abstract**

SHASTRA [1] is a collaborative distributed geometric design and manipulation environment. In this software project we consider the research and development of the next generation of software environments where multiple users (say, a collaborative engineering design team) create, share, manipulate, simulate, and visualize complex three dimensional geometric designs over a distributed heterogeneous network of workstations and supercomputers. SHASTRA consists of a growing set of tools for geometric design networked into a highly extensible environment where all the toolkits are interoperable. It provides a unified framework for collaboration, session management, data communication and graphical command interface along with a powerful numeric, symbolic and graphics substrate, enabling the rapid prototyping and development of efficient software tools for the creation, manipulation and visualization of multi-dimensional geometric data. These software tools are primarily tailored for use in both scientific experimentation and education, with an emphasis on distributed and collaborative geometric problem solving.

## 1 Introduction

SHASTRA is a highly extensible, collaborative, distributed geometric design and manipulation environment. At its core is a powerful collaboration substrate to support multi-user applications, and a distribution substrate which emphasizes distributed problem solving. Under the umbrella of Project SHASTRA, we have developed software toolkits GANITH[2], SHILP[3] and VAIDAK[4]. The GANITH algebraic geometry toolkit manipulates polynomial (i.e. algebraic) equations in any number of variables [9]. The SHILP modeling and display

*Supported in part by NSF Grant CCR 90-00028 and AFOSR-91-0276

[1] SHASTRA is the Sanskrit word for Science

[2] GANITH-SHASTRA is the Sanskrit word for Mathematical Science.

[3] SHILP stems from the Sanskrit word SHILP-SHASTRA for Sculpture.

[4] VAIDAK-SHASTRA is the Sanskrit word for Medical Science

toolkit manipulates curved solid objects with algebraic surface boundaries [2]. The VAIDAK medical imaging and model reconstruction toolkit manipulates medical image volume data [5]. Though each of these toolkits run as independent processes with separate user interfaces, they share a common infrastructure of numeric, symbolic, and graphics algorithms. The toolkit processes link to each other and communicate data structures (images, polynomials, solids, etc.) via inter-process communication facilities using an XDR-based protocol. SHASTRA provides these systems with connection management and data communication facilities enabling component systems to use facilities and operations provided by sibling systems, effectively integrating them into a large scientific manipulation system. It also provides them with a collaboration substrate to support cooperative and collaborative design effort.

SHASTRA provides a powerful geometric manipulation toolkit design substrate. It supports rapid prototyping of such systems emphasizing distributed design and collaboration. Abstracting away from the scientific manipulation aspect of its implementation, SHASTRA represents a paradigm for interoperable software tool design. The rest of this paper is structured as follows. Section 2 presents the design objectives of the SHASTRA system and then highlights the designed system's features contrasting prior related work. Section 3 details the distributed system architecture. Section 4 explains the collaboration infrastructure highlighting collaboration maintenance, the access regulation mechanism, and the multimedia aspect of our system. Details of SHASTRA groups and application specific support for collaboration are also presented. Specific details of our geometric design applications for distributed and collaborative problem solving are presented in sections 5 and 6.

# 2 SHASTRA – System Features

## 2.1 Design Objectives

As an extensible collaborative environment for geometric manipulation, the primary goals of SHASTRA are

1. Provide a rich substrate for design of collaborative tools.
2. Support design for interoperability.
3. Support distributed geometric design.
4. Device independence and portability.
5. Encourage code reuse.

Harnessing of current computing power has made it possible to tackle larger problems. Collaborative tools which permit multiple experts to cooperate on designs and collectively solve problems are the tools of the future. This is a step ahead of systems that support multiple users in a distributed design setting. In the case of large designs it will be much

more productive to allow multiple users to cooperate on a design. SHASTRA incorporates, into its core, a powerful collaboration substrate which can be used to create powerful objective-specific multi-user applications.

Geometric design and manipulation systems inherently have a certain amount of commonality. They all need display and visualization facilities, algebraic manipulation tools etc. In a large scientific manipulation environment, it is very productive to be able to use facilities provided by one toolkit within another. SHASTRA presents a design-for-integration paradigm for design of geometric manipulation tools which stresses on code reusability and interoperability with other systems. The functionality of the new toolkit is not directly relevant. SHASTRA suggests an architecture for the system which allows it to use a powerful substrate of existing low-level facilities , and at the same time, easily integrating it with existing systems. Toolkits designed around the SHASTRA-supported paradigm will therefore be able to use facilities from other toolkits while offering their own functionality.

We believe that a loosely coupled system of highly function specific tools is better than giant systems which aim to do everything. SHASTRA supports distributed geometric design. It stresses on interoperability of systems and on extensibility. The idea of a scientific manipulation environment is viable only if it is easy to integrate new systems into it. The distribution makes it easy to perform operations on hardware most suited for them.

SHASTRA stresses on device independence and portability of its component systems. We would like these tools to be able to run on a variety of hardware platforms, to be distributed in the true sense. This emphasizes the need for a standardized graphics interface to various graphics platforms. SHASTRA uses XS, a hardware independent graphics library suite, as its graphics platform. On similar lines, systems need to have a highly portable user interface. We use the X Window System [21], Version 11 Release 4 (subsequently referred to as X11 ) to develop user interfaces since it is supported by a large number of vendors. The toolkits are developed on Unix using C and Common Lisp.

To avoid wasted cycles, SHASTRA encourage code reuse by providing a rich substrate for systems being built under its umbrella. It makes polynomial manipulation, curve tracing, surface display, network connection management, data communication, collaboration management and other facilities available to such systems. The system designer incorporates these facilities into his design, and is able to integrate with the environment. This also facilitates adding existing systems to the SHASTRA environment with just a few modifications.

One of the major design goals of SHASTRA is to be its extensibility for incorporating future geometric design and manipulation toolkits. For example, we soon hope to add the toolkit **BHAUTIK** which incorporates programs for finite element PDE simulations, such as in the stress-analysis of medical solids created by VAIDAK. SHASTRA supports functionality sharing through its system integration facilities.

## 2.2 Highlights

SHASTRA provides a framework for the implementation of the Object–Multiple View–Multiple Controller paradigm for multi-user applications where information shared between

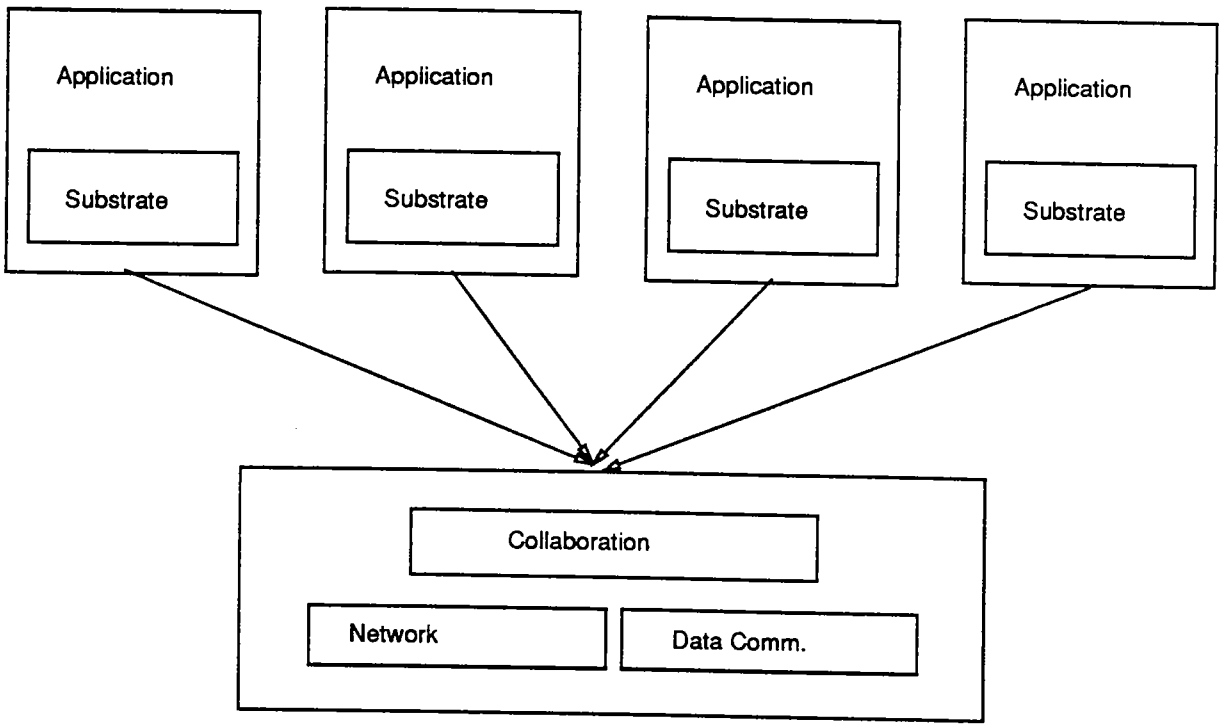


Figure 1: A SHASTRA Session

collaborating participants can be viewed and altered independently, but consistently, at the different sites. It provides a rich substrate for the development of distributed and collaborative applications by abstracting away the details of the underlying communication subsystem from the application developer. See Figure 1.

The SHASTRA architecture uses a replicated computation model for the multiple user system with a copy of the application running at each site involved in the collaboration. It provides collaborators with multiple channels of communication (currently text and graphics) to aid the collaboration effort. Centralizing the shared data in the session manager permits late joiners of ongoing sessions to be brought up-to-date immediately. The session manager also centralizes broadcasts alleviating the worries of misordering of input events which tend to create inconsistencies in distributed settings. SHASTRA provides a framework for specifying a constraint management subsystem which can be adapted to diverse applications. The environment promotes rapid prototyping of multi-user collaborative tools.

The distributed aspect of SHASTRA, with its emphasis on interoperability of tools provides a powerful environment for development of distributed problem solving applications. Once again, application developers adhering to the paradigm are relieved of the task of setting up a communication substrate. We use the X Window System (X11R4) as our

UIMS because of its widespread availability. The communication subsystem uses TCP as its reliable communication protocol. Multiple channel communication is effected using the select() facility in UNIX. Device dependence woes arising from machine representation are circumvented by using XDR to encode all information. We use the facility of rpcgen() to rapidly generate XDR encoding–decoding routines for different data structures.

The implementation of a scientific manipulation environment involves a large symbolic manipulation substrate which has been implemented in Common Lisp, which again makes it very portable. The LISP and C processes communicate via a protocol to remove the implementation dependence of foreign function calls. SHASTRA provides a rich substrate of graphics algorithms and decomposition techniques as well as a powerful numeric substrate as readily available C libraries.

The current implementation of SHASTRA runs on a mix of Sun, HP and SGI workstations.

## 2.3 Related Work

Previous research into collaborative sketching tools has resulted in many tools. The Capture Lab uses personal workstations with a large shared monitor, with one user controlling the monitor at a time [19]. Colab at Xerox PARC is a collaborative meeting facility which permits multiple users to simultaneously manipulate separate objects [22]. Xsketch is an X11 based tool which lets multiple users share two dimensional sketches across the Internet [18]. It suffers from limitations in richness of easily expressed sketches, and the tool is very application specific.

GROVE [14] and ShrEdit [4] are editors that are designed to support group editing of documents. Quilt is a group editor that lets multiple users work on different parts of the same document [15]. DistEdit [17] falls in the realm of toolkits for building collaborative applications – it supports building of interactive group editors by minor alterations to existing ones. However, it implements a master-slave concept, not allowing simultaneous updates from different participating editors.

LIZA is a collection of high-level tools for rapid groupware prototyping providing support for group editing and message transmission etc. [16].

ICICLE is a code inspection environment [12]. However it supports only master-slave blackboarding of the code window.

We considered the ISIS system for use as the communication substrate, since it has elegant broadcast facilities [10]. However, we chose to develop our own communication subsystem, with a well defined functional interface, so that we could test other communication models and special protocols for reliable broadcast.

Finally, Rendezvous proposes a powerful architecture for multi-user applications [20]. However, it fails to address the issue of adapting existing applications to multi-user versions.

This work does not focus on systems for sharing windows or viewing surfaces among multiple users which provide content independent sharing. Though such schemes are very suited for electronic blackboarding, they are not good tools for multi-user interaction since they are completely removed from the aspect of sharing data.

### 2.4 Application Potential

SHASTRA is a very powerful distributed and collaborative toolkit prototyping environment. It provides a rich substrate for design of such systems. Though a scientific manipulation environment has been the focus of our implementation, the facilities easily abstract out to a variety of situations requiring similar substrates.

The collaborative layer is generic and can be used to implement the heart of systems for collaborative editing, code viewing and quality assurance tools, software development environments, multi-user CAD tools, and interactive multi-player games etc.

The distribution aspect lets us build sophisticated problem solving environments where powerful tools are interoperable with other powerful tools. The integration of 3-D graphics into the environment adds a new dimension to the potency of this environment, as visual processing on sophisticated graphics engines becomes more common.

## 3 System Architecture

### 3.1 SHASTRA

The SHASTRA kernel spawns all needed design toolkit processes, manages these processes, and provides for all interprocess communication of the data structures that must be passed between toolkits. SHASTRA runs at a well known port on a host. This ensures that there is at most one instance of SHASTRA per host, and also provides remote systems a handle on the process. SHASTRA is responsible for maintaining all information relevant to the collaborative environment. It keeps track of active toolkits on the local host, and remote SHASTRAs which comprise the distributed environment. See Figure 2.

The front end to SHASTRA exists on a per-user basis, and is used to start up component systems locally or remotely, initiate collaborations and permit joining/leaving ongoing collaborative sessions. Systems can also connect directly to each other via SHASTRA to exchange data outside a collaborative setting. This is primarily to support distributed problem solving in the SHASTRA environment in single user mode.

### 3.2 COLLAB

The COLLAB library provides the application with support for multi-user collaborations. The suite of collaboration management routines provides SHASTRA applications with a straightforward interface to collaborative sessions, with facilities for initiating, terminating, joining, leaving and conducting collaborations.

A Session Manager (SAHYOG) runs per collaborative session. It maintains the collaboration and handles details of connection and session management. It is a repository of the objects in the collaboration, and keeps track of membership of the collaborative group. It has a constraint management subsystem which resolves conflicts that arise as a result of multi-user interaction, and therefore maintains mutual consistency of multiple operations. The session manager provides the broadcast facility needed for electronic blackboarding.

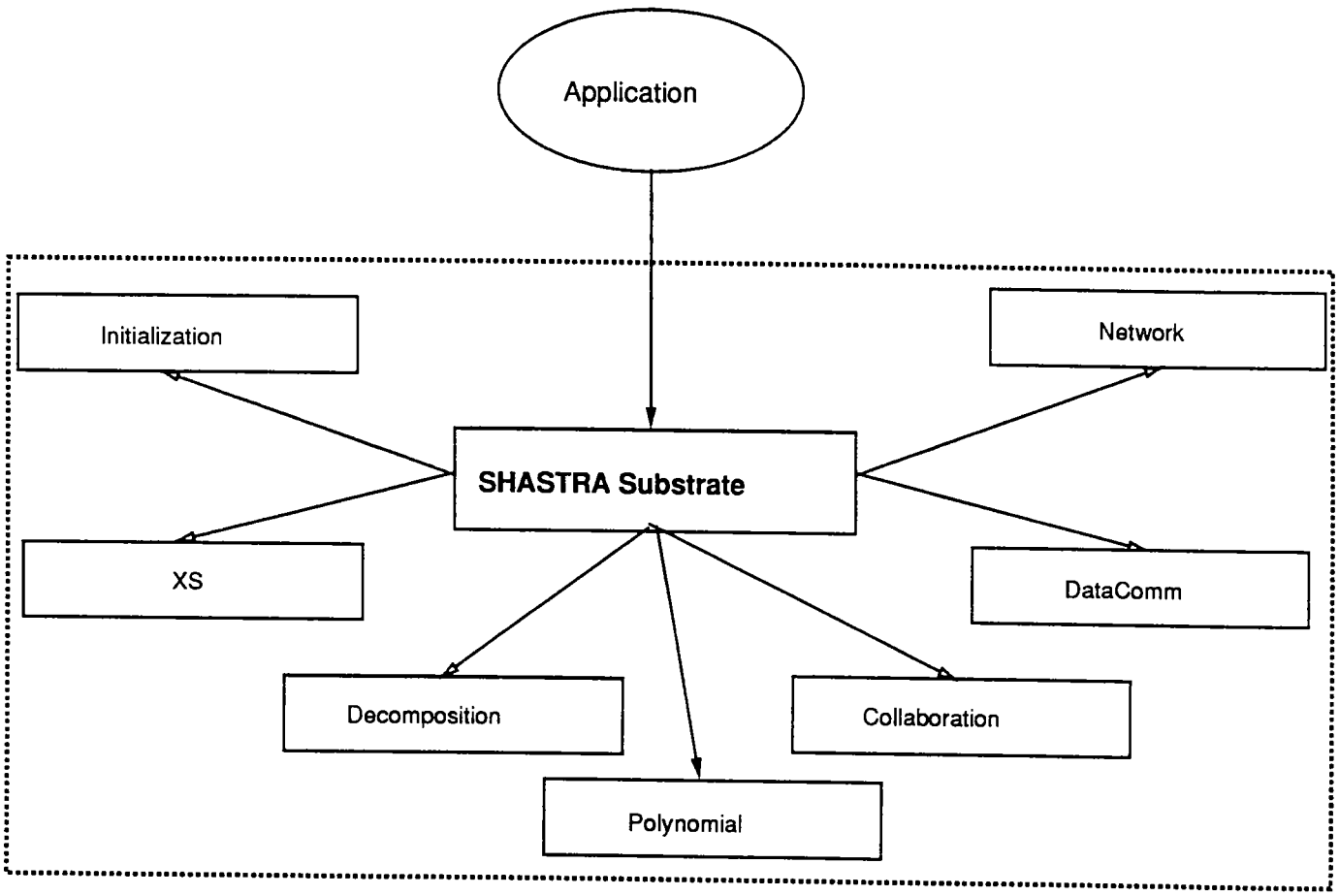


Figure 2: The SHASTRA System Architecture

Every toolkit participating in the collaboration communicates directly with the session manager. It registers collaboration objects with the manager and transmits to it all operations it performs on these objects. The session manager performs constraint checks and relays requisite information to all other participants in the collaboration.

A collaborating system will perform certain operations on behalf of the session manager. Every system thus executes in two modes – local mode, when it responds to directives from the user interface and remote mode, when it responds to directives from remote systems (the session manager, in this case). A system may be party to multiple collaborative sessions simultaneously.

The SAHYOG system also provides a blackboarding substrate to collaborating systems. This is an asymmetric operation in which one [redacted] system is master and all others are slaves. A token passing scheme is used to select a master. All operations performed by the master are relayed to the slaves. This facility is very useful for instructional purposes.

## 3.3 NETWORK

The Network library is a collection of connection management routines which permit an application to link up with SHASTRA, the user interface, back end processes and other systems with which the application communicates. This library provides a mechanism to handle multiple channels by multiplexing them. The library also provides templates for setting up clients for the functionality desired from other systems, and servers for the functionality offered to other systems by the application. All communication in this socket based system is effected through reliable protocols. The NETWORK layer is the heart of the extensible distributed environment.

## 3.4 DATACOMM

The Data Communication library is a mechanism to facilitate exchange and sharing of data structures between different applications or between different instances of the same application. DATACOMM provides a device independent, hierarchical data structure transfer system using the XDR translation facilities provided by "rpcgen". This is an important part of the SHASTRA substrate, as it facilitates extensibility of the data structure set in the system. The library uses an XDR based protocol to implement the inter-process communication. The use of XDR makes it possible to achieve true heterogeneity for the data communication substrate, as it makes it portable across different machine representations. This permits a richer hardware platform mix in the SHASTRA environment. The hierarchical design of the data transfer mechanism permits efficient communication of changes between communicating systems.

## 3.5 XS

The XS Graphics Libraries are a suite of 3D graphics libraries that access system-*dependent* graphics facilities (and hardware) in a uniform, system-*independent* manner [3]. Each system

supported is represented by a single library in the suite. All libraries in the suite present the same function-call interface. In this way, an application program can maintain source-level portability across several systems by simply linking with appropriate members of the XS suite. The current members of the XS suite consist of the X Window System (X11 Release 4) on SUN workstations and other X11 platforms, the GL graphics library specialized for SGI workstations, the STARBASE graphics library on HP systems and Windows 3.0 on IBM compatible PCs. The XS library permits powerful device independent 3D graphics in the SHASTRA environment, permitting component systems to execute on a variety of graphics platforms. XS de-links the system developer from concerns about the specifics of multiple graphics engines. SHASTRA can thus run on a heterogeneous hardware platform.

### 3.6 Poly Package

The Polynomial package is a library for numeric and symbolic manipulation of polynomials. It provides a set of routines for addition, subtraction, multiplication, division, differentiation of multivariate polynomials [7]. The polynomial package is used for algebraic surface computations, and for display of such surfaces [6].

### 3.7 Decomposition Package

The Decomposition Package takes curve and surface patches and produces piecewise linear approximations which are suitable for display. The package decomposes algebraic equations to sets of line segments and polygons which can be passed on to a graphics pipeline for visualization.

The Decomposition Package handles algebraic curves and surfaces, Bernstein curves, Bernstein-quad surfaces, and Bernstein-tensor surfaces [6, 8]. The numerical curve tracer allows the display of edges defined by the intersection of two arbitrary algebraic surfaces delimited by two vertices which are assumed to lie on both surfaces.

## 4 The Collaboration Infrastructure

The SHASTRA architecture uses a replicated computation model for the multiple user system – a copy of the application runs at each site involved in the collaboration. This delivers a substantial performance advantage over a centralized model. The architecture manages only collaboration specific windows of the application. This, coupled with the replicated model permits easy separation of the private and public aspect of the application at each site.

### 4.1 Application Architecture

All applications designed to run in the SHASTRA environment have certain features which make them amenable to inter-operation. A typical application has an application specific core – the Application Engine which implements all the functionality offered by the system

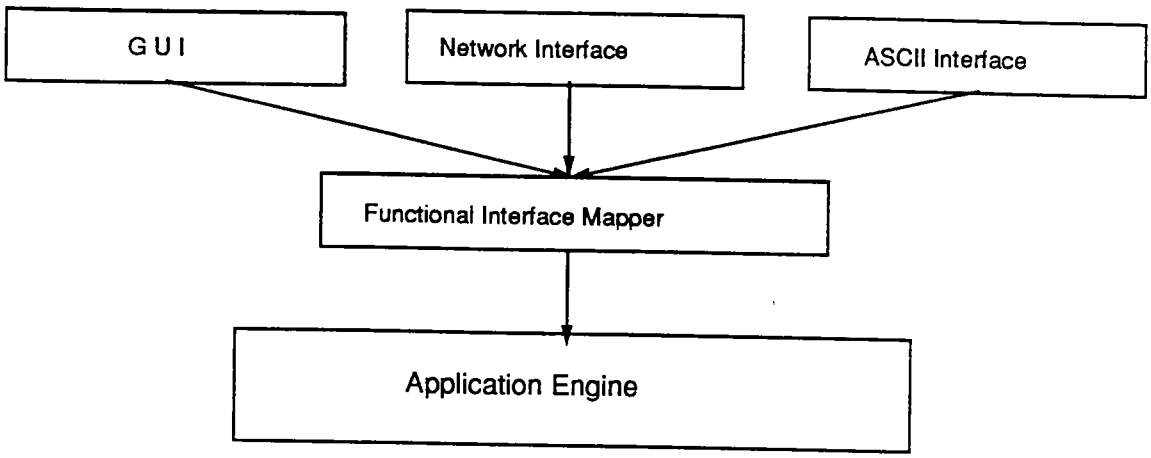


Figure 3: The Architecture of a SHASTRA Application

as a tool. In our case, these are usually scientific manipulation and solid modeling tools, but functionality is in no way restricted to this domain. On top of the Engine is a Functional Interface Mapper which actually calls upon functionality embedded in the engine in response to requests from the the Graphical User Interface, ASCII Interface or the Network Interface. See Figure 3.

This architecture makes it easy for other systems to connect to the application via the network interface and request operations. Applications can support multiple network interfaces simultaneously, using the multiplexing facility provided by the NETWORK substrate.

Applications have access to all the functionality of the COLLAB library. which provides them with support for multi-user collaborations – a straightforward interface to collaborative sessions, with facilities for initiating, terminating, joining, leaving and conducting collaborations.

## 4.2 Collaboration Maintenance

A collaboration is initiated through the SHASTRA shell by one of the applications, by specifying the participants, and the capabilities they will have during the collaboration. SHASTRA contacts the respective systems and adds them to the collaborative session if they agree to participate in the collaboration. Applications can dynamically discover existing SHASTRA systems by using an Oracle with which every SHASTRA process registers on instantiation.

Collaboration is performed under the auspices of a Session Manager (SAHYOG) One manager runs per collaborative session. It maintains the collaboration and handles details of connection and session management. It is a repository of the objects in the collaboration, and keeps track of membership of the collaborative group. It has a constraint management

subsystem which resolves conflicts that arise as a result of multi-user interaction, and therefore maintains mutual consistency of multiple operations. Constraint management is fairly straightforward since there is only one site performing the arbitration. The session manager also provides the broadcast facility needed for electronic blackboarding.

Every toolkit participating in the collaboration communicates directly with the session manager. It registers collaboration objects with the manager and transmits to it all operations it performs on these objects. The session manager performs constraint checks and relays requisite information to all other participants in the collaboration.

A collaborating system will perform certain operations on behalf of the session manager. Every system thus executes in two modes – local mode, when it responds to directives from the user interface and remote mode, when it responds to directives from remote systems (the session manager, in this case). A system may be party to multiple collaborative sessions simultaneously.

The SAHYOG system also provides a blackboarding substrate to collaborating systems. This is an asymmetric operation in which one one system is master and all others are slaves. A token passing scheme is used to select a master. Identity of the master is broadcast to all collaborators. As long as he has locked the token, his actions are broadcast to the rest of the group. There is no preemption, but other participants can request the master to release the token. Token passing is performed in software, between SHASTRA front-ends. This guarantees that the token will not get stuck if any participant gets computation bound. The blackboarding facility is very useful for instructional purposes, demonstrations and walk-throughs.

### 4.3 Access Regulation Mechanism

A permissions based system defines the scope of collaboration operations available at each participant. Permissions are maintained at the Session Manager which centrally coordinates the access regulation mechanism. The session initiator specifies permissions at the start of the collaboration. Permissions are specifiable on a per-participant as well as a per-object basis. It is intended to be a regulatory mechanism, rather than a security mechanism. The mechanism provides only an implementation, and issues of policy are left unspecified. The regulatory subsystem involves the following kinds of permissions.

1. Access – This regulates the view at a collaborative site. For a participant, it specifies whether or not he will observe blackboarding type operations. For an object it specifies whether or not it will appear as part of the view.

2. Browse – This permission controls whether the site can browse through objects independently, e.g. in the current scenario of modeling, independent control of viewing transformations is regulated by this permission. In the setting of collaborative editing, it controls whether or not a collaborator can perform a read-only browse of the document. The site can toggle between synchronous and asynchronous browse modes which switch it between slave-mode and independent mode.

3. Modify – This controls participation in a collaboration. Only the sites with this permission set can collaborate through the session manager.

4. Copy – This permission regulates copy propagation. It permits sites to obtain a private copy of the collaborative object and perform independent design or modeling operations and/or alterations. The copied object becomes a local object until it is explicitly reintroduced into the collaboration.

5. Grant – This permission makes sense only in the context of participants of the collaboration session. The initiator of the session grants this permission to other trusted sites which can subsequently grant permissions to other members. This effectively distributes the task of granting permissions at run-time.

In general, per-object permissions, when defined, take priority over per-site permissions. The Modify permission is an exception, where the per-site permission takes priority. Also, since permissions can be altered dynamically from multiple sites, the last permission granted takes precedence.

The regulatory mechanism aims at being able to support a variety of interactions ranging form master-slave mode blackboarding to multiple site collaboration.

### 4.4 The Multimedia Aspect

Current functionality in the collaborative geometric manipulation system supports simultaneous text and graphics channels for information transmission during the collaboration. We also use an externally conducted telephonic conference to aid the collaboration. We are investigating building in audio and video channel support into the collaborative substrate, though it has not been a focus area with us thus far.

### 4.5 Application Support for Collaboration

All the applications integrated into SHASTRA's collaborative environment so far predate the collaborative system. It has been important, therefore, to create interfaces which permit upgrading existing systems with minimal changes so that they would inter-operate with other systems under the SHASTRA umbrella. In particular, the issues of Application Architecture, Collaboration Maintenance and Access Regulation are of concern. Applications need to be modified to adhere to the specified architecture, which now serves as a blueprint for design of new ones. Applications interface to the collaboration subsystem via functionality provided in the COLLAB library – facilities for initiating, terminating, joining, leaving and conducting collaborations. The application performs display and maintains objects of the collaboration on behalf of SHASTRA. It informs the session manager whenever operations are performed, and updates its information whenever directed to do so by the session manager. The session manager requests all participating systems to create a window on its behalf and regulates what goes on in that window. The paradigm causes all endpoints of a collaboration to be symmetric in the sense that all sites must run instances of the same

applications. This facilitates hiding the details of the application from the collaborative system, making it usable in completely different settings.

### 4.6 Shastra Group Concept

The concept of a group emerges in the scenario of a collaboration. Collaborative tasks are conducted between members of a collaborative group. A group is typically created by the initiator of a collaboration at startup. He also becomes the group leader, and is in charge of additions to and deletions from the group. Group information is maintained by the SHASTRA kernel, which stores primarily the following information about all systems that run under it:

1. Host-id – Internet address of host.
2. User-id – of application process.
3. Port-id – of the application, where it can be contacted.
4. System-id – to Identify the kind of SHASTRA subsystem.
5. Display-id – specifying the display used for the user interface.
6. Group-ids – specifying membership in currently active collaborative groups.

Thus SHASTRA serves as a repository of this contact information. Any remote system aiming to contact extant systems for a collaborative session retrieves information about SHASTRAs from an Oracle, and then gets system specific contact information from the respective SHASTRA.

To join a group, such a system requests grant of membership from the SHASTRA that holds the group, which in turn gets permission from the group leader. To discontinue membership from a group, either the system requests the host SHASTRA to delete it from the group, or the group leader deletes the system from the group. In case of failure of the group leader, SHASTRA promotes one of the collaborators to the position of leader.

### 4.7 Information Flow

A well known Oracle dynamically stores information about the existence of different SHASTRAs. SHASTRA is a repository for contact information for ongoing collaborations and other concurrently executing applications running under it. It maintains a control link to all these sub-systems.

A session manager connects to its host SHASTRA, and maintains data links to all the members of the collaboration it is conducting. It also serves as a repository for objects in the collaboration. It is at the hub of a star topology. This setup tends to suffer from performance degradation when the number of collaborators is large, but works well for typical collaborative groups. We are looking at reliable broadcast protocols, and multicast

mechanisms to alleviate this problem. We currently simulate broadcast using point-to-point transmissions.

In a non-collaborative setting for distributed problem solving, applications connect directly to each other, using the contact information stored by SHASTRA, to exchange data and utilize functionality offered.

All communication is done using TCP connections, and data transfers are performed via an XDR based protocol.

# 5 SHASTRA Applications

The SHASTRA scientific manipulation environment currently includes the GANITH algebraic geometry toolkit, the SHILP solid modeling and display toolkit, and the VAIDAK medical imaging and model reconstruction toolkit.

## 5.1 GANITH

GANITH can be used to solve systems of algebraic equations and visualize its multiple solutions. Example applications of this are curve and surface display, curve-curve intersections, surface-surface intersections, curve-surface intersections, etc. A graphical user interface allows the display and animation of 0-dimensional (points), 1-dimensional (curves) or 2-dimensional (surfaces) solutions. The software has two major components. A Graphical User Interface, written in C and X11, and a Computer Algebra Substrate, written in Lisp. Each software component resides in a Unix process, and they communicate using a special string-based protocol over the pipe inter-process communication mechanism. The color rendering and shading utilities have been developed in C for the HP and IRIS workstations using the STARBASE and IRIS graphics libraries, respectively. The user interface under X11 is mostly menu based and user friendly. GANITH shall prove useful for various undergraduate and graduate education courses in calculus, analytic geometry and algebraic geometry. It would also prove invaluable to researchers in the areas of algebraic geometry, geometric modeling and computer graphics.

## 5.2 SHILP

SHILP can be used for the geometric design (creation, editing, etc.) and display of solid models with algebraic surfaces. Curves and surfaces can be represented in both implicit and rational parametric form, in either power or Bernstein polynomial bases. The current functionality of the toolkit includes restricted extrude, revolve and offset operations, edit operations on planar lamina and polyhedral solids, fleshing of wireframes with interpolating surfaces, and color rendering of solids. Three dimensional grids or meshes superimposed on solid objects can also be generated and are used for finite difference solutions of partial differential equations. For the purpose of finite element computation, algorithms have been implemented which decompose arbitrary polyhedra with holes into convex pieces or tetrahedra. The software has three major components. A Graphical User Interface, written in C

and X11, a Lamina Manipulation Substrate for the creation and editing of two dimensional curved facets written also in C and a Solids Modeling Substrate for solid object creation, editing and decomposition, written in C and Lisp. Each software component resides in a Unix process, and they communicate using a special string-based protocol over inter-process communication. The color rendering and shading utilities as part of the Graphical User Interface have been developed in C for the HP and IRIS workstations using the STARBASE and IRIS graphics libraries, respectively. SHILP has been used an education tool for graduate courses on geometric modeling and computer graphics. It is also an experimental test bed for new algorithms being developed for geometric design and graphics display and animation of complex curved objects.

### 5.3 VAIDAK

VAIDAK can be used to construct accurate surface and solid models of skeletal and soft tissue structures from CT (Computed tomography), MRI (magnetic resonance imaging) or Laser surface imaging data. VAIDAK incorporates both heuristic and exact methods of contouring image data, active thresholding, tiling or polygon reconstruction, shading and rendering solid models. It also incorporates a browser feature to modify the contours, a scanner to view image data and interactively pick threshold values, and a render window to change lighting and display modes. VAIDAK was initially implemented in C on a Silicon Graphics workstation under the NEWS windowing system. Its interface and software tools are now being modified to run in C with an X11 window interface. The software again has three major components. A Graphical User Interface, to be written in C and X11, a Contour Manipulation Substrate written in C and which is shared with the Lamina Manipulation Substrate of SHILP, and a Solids Modeling Substrate for solid model reconstruction written in C and common to SHILP. The color rendering utilities as part of the Graphical User Interface are also being developed in C for the HP and IRIS workstations. VAIDAK shows great promise as a scientific education and experimentation tool for medical practitioners, surgeons and scientists involved in artificial limb design.

## 6 Distributed and Collaborative Problem Solving

The SHASTRA environment has been designed to promote Distributed and Collaborative Problem Solving by providing a rich set of interoperable tools in a powerful user friendly setting. As more tools are integrated into the environment, it will be possible to perform highly sophisticated scientific manipulations. In the setting of available functionality, human skeleton modeling is a good example of distributed problem solving. CT/MRI data is input to the medical imaging system to produce a polyhedral solid in VAIDAK. This is passed on to a SHASTRA assisted instantiation of SHILP. See Figure 4.

SHILP calls upon instances of GANITH to interpolate the polyhedral surface and produce compact curved surface solid models of the skeleton. See Figure 5.

The above scenario can be used to construct an application for interactive design of arti-

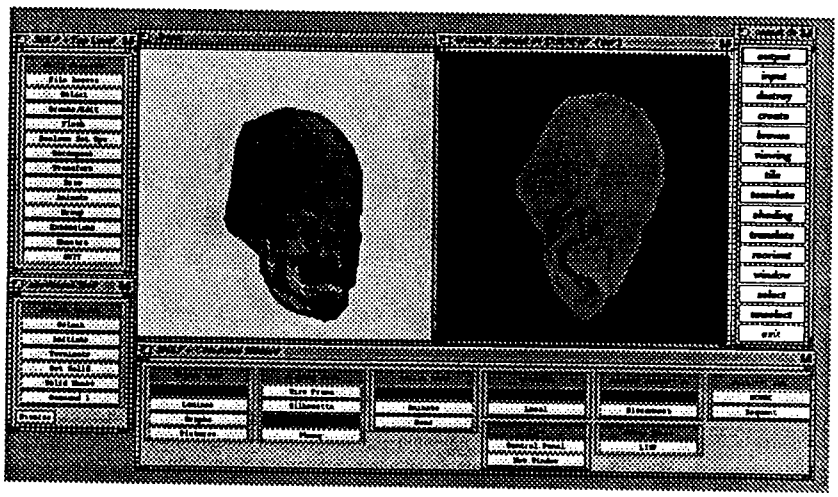

Figure 4: Data Structure Communication between VAIDAK and SHILP

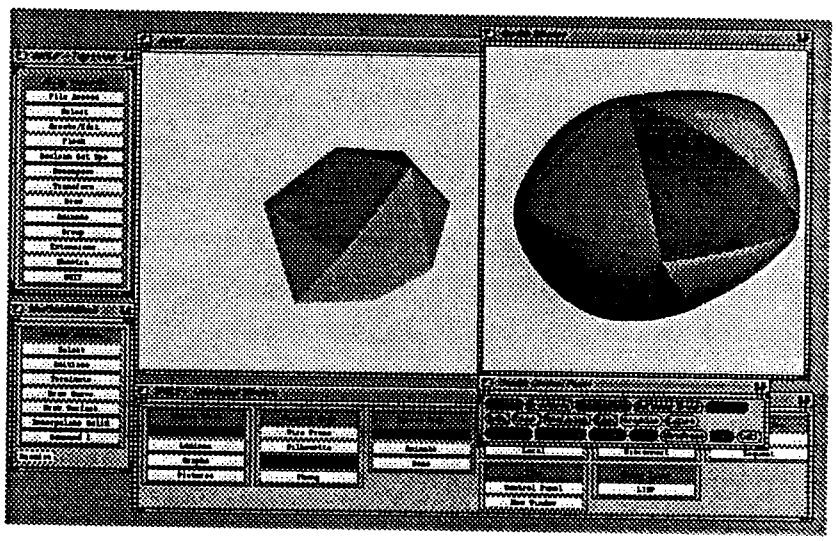

Figure 5: A Polyhedron Smoothed in SHILP via Remote Calls to GANITH

ficial implants using SHILP which makes remote calls to GANITH for smoothing operations. This smoothing operations has been implemented as a distributed algorithm which allows a user of SHILP to instantiate a GANITH process and a remote call, separately for each face of the polyhedron. Substantial speedup is thereby obtained for complex designs via this distributed parallelism. In a SHASTRA collaborative session multiple users can share the same skeleton solid or artificial implant model and interactively work on geometrically disjoint sub-parts of the same solid - thereby achieving even faster design throughput.

# 7 Issues

Currently, when a user wants to request operations on remote systems, he needs to direct SHASTRA about where the operation will be performed. The user explicitly specifies which remote system to connect to. It would be useful for SHASTRA to be able to assign tasks to new instances of server systems, using runtime load analysis to distribute the computation to less busy machines. It will still be useful to retain user defined over-ride facilities. SHASTRA will then also function as a task-broker for distribution of scientific computations.

SHASTRA has been designed with the issue of extensibility being given prominence. However, in this design, though systems use SHASTRA to establish communication channels with other systems, the final network connections are established between the communicating systems. In large design problems, it is possible to achieve a star topology, with a client system at the hub, and server systems at the surrounding nodes. The possibility of performance degradation due to high connection density may require us alter our connection scheme. However, since most of the design operations currently are computation-intensive, it is not a major issue at this time.

# 8 Conclusions

SHASTRA meets its design objectives as an extensible collaborative environment for geometric manipulation. The ease of integration of GANITH, SHILP and VAIDAK into the SHASTRA system demonstrates that the idea is very viable. The substrate provided by SHASTRA permits the systems to run on a variety of hardware platforms , meeting the device independence and portability requirement without compromising on utilization of powerful resident graphics facilities when they are available. Though the current component systems predate SHASTRA, it has not been hard to modify them to be integrable into its environment. The design for integration paradigm supported by SHASTRA will make the task even easier. We are in the process of adding BHAUTIK, a finite element system, under the SHASTRA umbrella. Integration of available subsystems into a powerful substrate has helped promote code reuse. SHASTRA is a veritable software laboratory, which we currently use as a test bed for a variety of graphics algorithms and solid modeling techniques. SHASTRA is being used to assist in teaching undergrad and grad level courses in Computer Science.

## References


[1] Abdel-Wahab, H., Guan, S., Nievergelt, J., (1988) " Shared Workspaces for Group Collaboration: An Experiment using Internet and Unix Inter-process Communication", *IEEE Communications Magazine*, 10-16.

[2] Anupam, V., Bajaj, C., Dey, T., and Ihm, I., (1991), "The SHILP Solid Modeling and Display Toolkit", CAPO Technical Report 91-29, Computer Science Department, Purdue University.

[3] Anupam, V., Bajaj, C., Burnett, A., Fields, M., Royappa, A., Schikore, D., (1991), "XS: A Hardware Independent Graphics and Windows Library", CAPO Technical Report 91-28, Computer Science Department, Purdue University.

[4] Cognitive Science and Machine Intelligence Lab, U. of Michigan, Ann Arbor, "ShrEdit 1.0: A Shared Editor for the Apple Macintosh, User's Guide and Technical Description", 1990.

[5] Bailey,B., Bajaj, C., and Fields, M., (1991), "The VAIDAK medical imaging and model reconstruction toolkit", CAPO Technical Report 91-31, Computer Science Department, Purdue University.

[6] Bajaj, C., (1988), "Geometric Modeling with Algebraic Surfaces", *The Mathematics of Surfaces III*, ed. D. Handscomb, Oxford University Press, 3–48.

[7] Bajaj, C., (1990), "Geometric Computations with Algebraic Varieties of Bounded Degree", in *Proceedings of the Sixth ACM Symposium on Computational Geometry*, Berkeley, California, 148–156.

[8] Bajaj. C., (1991), "Surface Fitting with Implicit Algebraic Surface Patches", *Curve and Surface Modeling*, edited by H. Hagen, SIAM Publications, 1991, in press.

[9] Bajaj, C., and Royappa, A., (1989), "The GANITH Algebraic Geometry Toolkit", in *Proceedings of the First International Symposium on the Design and Implementation of Symbolic Computation Systems, Lecture Notes in Computer Science*, No. 429, Springer-Verlag (1990), 268–269. Also as a CAPO Technical Report 91-30, Computer Science Department, Purdue University.

[10] Birman, K., Cooper, R., Joseph, T., Kane, K., Schmuck, F., (1989) "The ISIS System Manual", June 1989.

[11] Bly, S., (1988) "A Use of Drawing Surfaces in Different Collaborative Settings", *Proc. Conf. on Computer-Supported Cooperative Work*, 250-256.

[12] Brothers, L., Sembugamoorthy, V., Muller, M., (1990) "ICICLE: Groupware For Code Inspection", *Proc. ACM CSCW 90*, 169-181.

[13] Crowley, T., Milazzo, P., Baker, E., Forsdick, H., Tomlinson, R., (1990) "MMConf: An Infrastructure for Building Shared Multimedia Applications", *Proc. ACM CSCW 90*, 329-342.

[14] Ellis, C., Gibbs, S., Rein, S., (1988) "Design and Use of a Group Editor", *Report STP-263-88*, MCC Software Technology Program.

[15] Fish, R., Kraut, R., Leland, M., Cohen, M., (1988) "Quilt: A Collaborative Tool for Cooperative Writing", *Proc. ACM SIGOIS Conference*, 30-37.

[16] Gibbs, S., (1988) "LIZA : An Extensible Groupware Toolkit", *Report STP-042-88*, MCC Software Technology Program.

[17] Knister, M., Prakash, A., (1990) "DistEdit: A Distributed Toolkit for Supporting Multiple Group Editors", *Proc. ACM CSCW 90*, 343-355.

[18] Lee, J., (1990), "Xsketch : A Multi-user Sketching Tool for X11", *ACM Office Information Systems*, 169-173.

[19] Mantei, M., (1988) "Capturing the Capture Concepts: A Case Study in the Design of Computer-Supported Meeting Environments", *Proc. Conf. on Computer-Supported Cooperative Work*, 257-270.

[20] Patterson, J., Hill, R., Rohall, S., Meeks, M., (1990) "Rendezvous: An Architecture for Synchronous Multi-User Applications", *Proc. ACM CSCW 90*, 317-328.

[21] Scheifler, R., Gettys, J., Newman, R., (1986) "The X Window System", *ACM Transactions on Graphics*, 5, 2, 79-109.

[22] Stefik, M., Foster, G., Bobrow, D., Kahn, K., Lanning, S., Suchman, L., (1987) "Beyond the Chalkborad: Computer Support for Collaboration and Problem Solving in Meetings", *Comm. of the ACM*, 30, (1), 32-47.